OFF ONE 2022

Reverse engineering an IP camera DAHUA

Васин Юрий

Специалист «Raccoon Security»

@y0v1737

Moscow, August 25-26, 2022

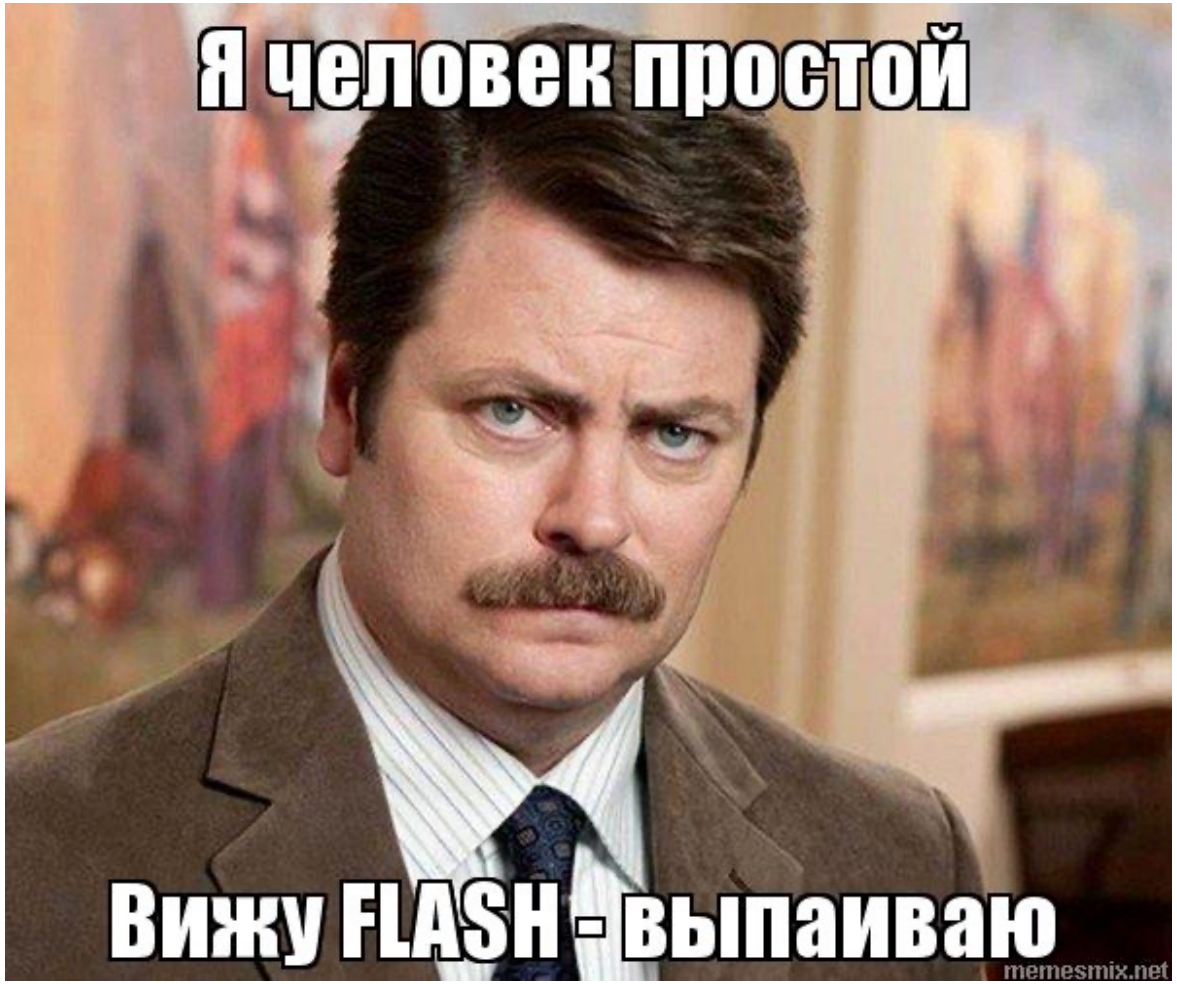
AGENDA

- Getting firmware
- Unpack firmware
- Decrypt firmware
- Their manners
- Shell with QR
- Root of trust
- Questions

GET FIRMWARE

GETTING FW

Try binwalk

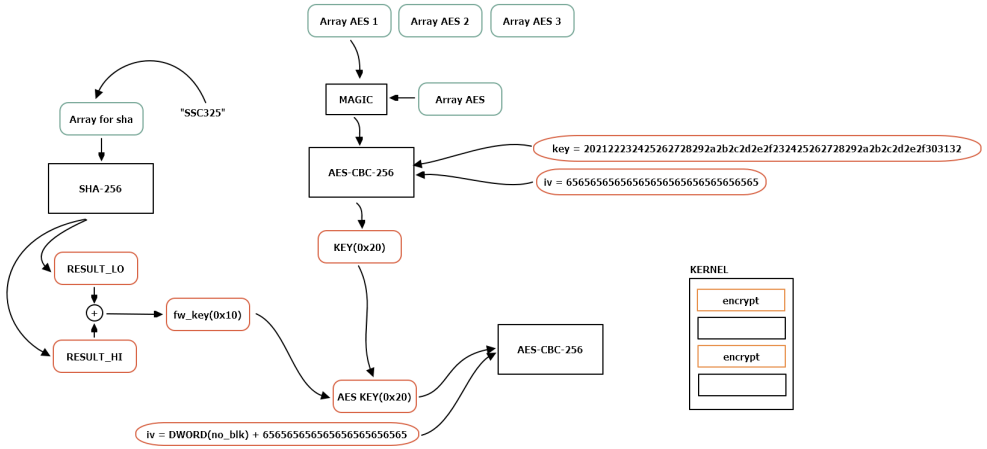


Partitions
_uImg.part.extracted
_uImg.part2.extracted
_uImg.romfs.extracted
_uImg.romfs2.extracted
_uImg.uboot.extracted
_uImg.web.extracted
_uImg.web2.extracted
uImg.kernel
uImg.kernel2
uImg.part
uImg.part2
uImg.romfs
uImg.romfs2
uImg.uboot
uImg.web
uImg.web2

PARTITIONV2.TXT

Name	Cs	Offset	Size	Mask_flags
mini-Boot	0	0x0000000000000000	0x0000000000280000	RW
U-Boot	0	0x0000000000340000	0x00000000000C0000	RW
hwid	0	0x0000000000400000	0x0000000000100000	RW
dgs	0	0x0000000000500000	0x0000000000100000	RW
syslog	0	0x0000000000600000	0x0000000000700000	RW
config	0	0x0000000000D00000	0x0000000000700000	RW
backup	0	0x0000000001400000	0x0000000000700000	RW
updateflag	0	0x0000000001B00000	0x0000000000100000	RW
partition	0	0x0000000001C00000	0x0000000000100000	RW
product	0	0x0000000001D00000	0x0000000000500000	RW
Kernel	0	0x0000000002200000	0x0000000000700000	RW
web	0	0x0000000002900000	0x0000000000A00000	RW
romfs	0	0x0000000003300000	0x0000000001900000	RW
aewb	0	0x0000000004C00000	0x0000000000200000	RW
backpart	0	0x0000000004E00000	0x0000000000100000	RW
backproduct	0	0x0000000004F00000	0x0000000000500000	RW
backkernel	0	0x0000000005400000	0x0000000000700000	RW
backweb	0	0x0000000005B00000	0x0000000000A00000	RW
backromfs	0	0x0000000006500000	0x0000000001900000	RW
backaewb	0	0x0000000007E00000	0x0000000000200000	RW

DECRYPT KERNEL



KERNEL

```
          0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F  0123456789ABCDEF
0000h: 1C 14 92 12 E5 FC 98 59 07 57 57 FE 15 0E CF 03  ..'.åü˜Y.WWþ..Ï.
0010h: F5 C4 FF D9 FE 33 7D F5 36 8F 71 DB 37 CC A3 A9  õÄÿÙþ3}õ6.qÛ7Ì£©
0020h: F1 20 18 22 E0 8C 15 F3 0A FB 7A 3D 6A 34 FC A0  ñ ."àŒ.ó.ûz=j4ü
0030h: 31 64 D3 19 FD 82 AB 09 C8 DB F6 85 5C 05 A2 F3  1dÓ.ý‚«.ÈÛö…\.¢ó
0040h: DB 1C 3C 9A 17 14 63 44 C7 A2 74 B6 E6 A7 15 E3  Û.<š..cDÇ¢t¶æ§.ã
0050h: C3 3F 52 83 08 7A 6A 8B 5B 3F 2E 14 08 D5 33 3B  Ã?Rƒ.zj‹[?...Õ3;
0060h: C0 B9 A3 4D 9C BB 73 E1 D8 47 63 B0 70 8F 4C EB  À¹£Mœ»sáØGc°p.Lë
0070h: 6A 9E 09 29 C3 BB A4 4F 0A 82 8A 4A 05 39 75 5E  jž.)Ã»¤O.‚ŠJ.9u^
0080h: 33 8E 82 B9 1D 0E FA 0E 81 B1 C7 14 49 41 6D 89  3Ž‚¹..ú..±Ç.IAm‰
0090h: AE 9A E1 D9 E8 C1 69 1E 44 63 49 44 C2 42 81 C6  ®šáÙèÁi.DcIDÂB.Æ
00A0h: FE E1 14 C8 45 47 3A 0F E9 86 DC D3 E2 ED 7C F3  þá.ÈEG:.é†ÜÓâí|ó
00B0h: 31 9C 5B AF DB D0 76 5E 70 13 CB E2 A1 85 B0 5E  1œ[¯ÛÐv^p.Ëâ¡…°^
00C0h: 5A 7B 04 3B 87 66 25 D4 29 A0 A6 52 3A B0 6C D6  Z{.;‡f%Ô) ¦R:°lÖ
00D0h: 45 49 49 0C C7 C0 D3 BA 0B 12 DA 35 BB 2B 9E 76  EII.ÇÀÓº..Ú5»+žv
00E0h: 96 2D 9F 2B 57 49 52 EB 17 13 87 47 C3 AF 29 A6  –-Ÿ+WIRë..‡GÃ¯)¦
00F0h: 86 2F AE F8 E0 C0 BF DE 81 0E 14 BB DC 4F 7A 26  †/®øàÀ¿Þ...»ÜOz&
0100h: 01 B2 9D 2D D7 CE 74 1B CE 84 C1 84 D2 B6 BE F5  .².-×Ît.Î„Á„Ò¶¾õ
0110h: 54 C2 6F 2A B3 C9 61 1F 14 7F 40 B8 BE C2 7E AF  TÂo*³Éa...@¸¾Â~¯
0120h: F8 56 95 FE DF 2E 92 87 F6 10 F4 BF 4C 76 50 1F  øV•þß.'‡ö.ô¿LvP.
0130h: CB C4 47 6F 09 F1 AB F0 7F BB 8F 91 16 C2 E2 ED  ËÄGo.ñ«ð.».'.Ââí
0140h: 60 26 4A 38 87 81 09 F5 46 1A B5 FC B6 C0 65 64  `&J8‡..õF.µü¶Àed
0150h: 0A B6 13 16 04 8C 1C 40 19 8F 19 E4 91 F5 F0 9F  .¶...Œ.@...ä'õðŸ
0160h: 3E F3 78 2C B7 0D 35 C7 B9 B2 DA 98 91 87 5D B3  >óx,·.5Ç¹²Ú˜'‡]³
0170h: 18 D8 4B BB B9 43 50 B4 77 F3 BD 31 D9 1E 77 09  .ØK»¹CP´wó½1Ù.w.
0180h: FD 14 19 E4 84 47 51 97 D0 1D BC 30 DC 99 96 62  ý..ä„GQ—Ð.¼0Ü™–b
0190h: CD B4 45 85 D4 73 20 0C DC 54 33 F4 50 F3 A7 F2  Í´E…Ôs .ÜT3ôPó§ò
01A0h: A4 10 BC F4 3A 6C 79 C2 46 72 37 90 5B C0 1E 40  ¤.¼ô:lyÂFr7.[À.@
01B0h: 08 FC 60 E7 AF 3D 11 06 58 77 EB 04 3C E8 F6 25  .ü`ç¯=..Xwë.<èö%
01C0h: 31 92 F2 99 00 8F 95 9A 29 72 E9 64 6E D6 25 D2  1'ò™..•š)rédnÖ%Ò
01D0h: 7F 9D C9 13 EE 10 71 9C B8 6E 58 2B 5F 59 CA 84  ..É.î.qœ¸nX+_YÊ„
01E0h: 50 71 99 A9 94 49 3C 76 B1 23 63 69 D3 EA 8B 09  Pq™©”I<v±#ciÓê‹.
01F0h: 9E 3C 38 CA 7B 5B ED 91 73 36 C7 B9 08 6C 4A BB  ž<8Ê{[í's6Ç¹.lJ»
```

```
0200h: 00 80 00 F0 1F 00 1A 28 06 D1 02 F0 75 F8 A0 EB  .€.ð...(.Ñ.ðuø ë
0210h: 05 00 50 44 02 F0 72 F8 A6 EB 05 09 09 F1 1F 09  ..PD.ðrø¦ë...ñ..
0220h: 29 F0 1F 09 09 EB 05 06 D1 44 36 E9 0F 5C AE 42  )ð...ë..ÑD6é.\®B
0230h: 29 E9 0F 5C F9 D8 A9 EB 06 06 B5 44 00 F0 70 FA  )é.\ùØ©ë..µD.ðpú
0240h: AF F2 43 10 30 44 87 46 50 EA 05 01 11 D0 83 44  ¯òC.0D‡FPê...ÐƒD
0250h: 84 44 02 44 03 44 DB F8 00 10 01 44 91 42 28 BF  „D.D.DÛø...D'B(¿
0260h: 8B 42 88 BF 49 19 4B F8 04 1B E3 45 F3 D3 2A 44  ‹Bˆ¿I.Kø..ãEóÓ*D
0270h: 2B 44 4F F0 00 00 42 F8 04 0B 42 F8 04 0B 42 F8  +DOð..Bø..Bø..Bø
0280h: 04 0B 42 F8 04 0B 9A 42 F5 D3 14 F0 01 0F 24 F0  ..Bø..šBõÓ.ð..$ð
0290h: 01 04 18 BF 00 F0 34 F8 20 46 69 46 0D F5 80 32  ...¿.ð4ø FiF.õ€2
02A0h: 3B 46 00 F0 63 FB 00 F0 3B FA 00 F0 F9 F9 39 46  ;F.ðcû.ð;ú.ðùù9F
02B0h: 42 46 FF F3 00 80 00 F0 1F 00 1A 28 40 F0 20 83  BFÿó.€.ð...(@ð ƒ
02C0h: 0F F2 10 0C DC F8 00 00 60 44 02 F0 17 F8 E0 F7  .ò..Üø..`D.ð.øà÷
02D0h: 00 80 FE E7 0C 06 00 00 98 02 00 00 F0 6F 26 00  .€þç....˜...ðo&.
02E0h: 0C 70 26 00 F0 6F 26 00 B4 6F 26 00 C4 6F 26 00  .p&.ðo&.´o&.Äo&.
02F0h: EC 6F 26 00 10 80 26 00 4C AF 36 00 AF F3 00 80  ìo&..€&.L¯6.¯ó.€
0300h: 4F F0 08 03 EF E0 4F F0 3F 00 06 EE 17 0F 06 EE  Oð..ïàOð?..î...î
0310h: 37 0F 4F F0 80 00 02 EE 10 0F 02 EE 30 0F 03 EE  7.Oð€..î...î0..î
0320h: 10 0F 4F F4 40 40 05 EE 30 0F 05 EE 10 0F 4F F0  ..Oô@@.î0..î..Oð
0330h: 00 00 07 EE 9A 0F 07 EE 15 0F 07 EE 16 0F 11 EE  ...îš..î...î...î
0340h: 10 0F 40 F0 2D 00 40 F4 80 50 01 EE 10 0F 4F F0  ..@ð-.@ô€P.î..Oð
0350h: 00 00 07 EE 15 0F 07 EE 16 0F F7 46 4F F0 3F 00  ...î...î..÷FOð?.
0360h: 06 EE 17 0F 4F F0 80 00 02 EE 10 0F 03 EE 10 0F  .î..Oð€..î...î..
0370h: 4F F4 40 40 05 EE 10 0F 4F F0 00 00 07 EE 10 0F  Oô@@.î..Oð...î..
0380h: 11 EE 10 0F 40 F0 0D 00 4F F0 00 00 01 EE 10 0F  .î..@ð..Oð...î..
0390h: 07 EE 10 0F F7 46 A4 F5 80 43 23 F0 FF 03 23 F4  .î..÷F¤õ€C#ðÿ.#ô
03A0h: 7C 53 18 46 4F EA 90 49 4F EA 89 49 09 F1 80 5A  |S.FOê.IOê‰I.ñ€Z
03B0h: 4F F0 12 01 41 F4 40 61 03 F5 80 42 49 45 28 BF  Oð..Aô@a.õ€BIE(¿
03C0h: 8A 45 21 F0 1C 01 34 BF 41 F0 10 01 31 43 40 F8  ŠE!ð..4¿Að..1C@ø
03D0h: 04 1B 01 F5 80 11 90 EA 02 0F EF D1 46 F0 04 01  ...õ€..ê..ïÑFð..
03E0h: 41 F4 40 61 7A 46 4F EA 12 52 41 EA 02 51 03 EB  Aô@azFOê.RAê.Q.ë
03F0h: 82 00 40 F8 04 1B 01 F5 80 11 01 60 F7 46 11 EE  ‚.@ø...õ€..`÷F.î
```

U-BOOT

```
print((int)off_2101BF14, "U-Boot 2010.06-svn8208 (Aug 04 2020 - 13:07:25)");
sub_2101B2B4();
sub_210173C4();
v0 = sub_2102C6EC();
v1 = dword_2101BF18;
if ( v0 )
{
  print2uart((const char **)"dgsBufStart:0x%08x\n", v0);
  *(_QWORD *)&v25[v1] = (unsigned int)v0;
}
```

U-BOOT: DECRYPT KERNEL

```
int __fastcall Bootm_handler(char **a1, int a2, int a3, int Args)
{
  int result; // r0

  result = bootm_try(a1, a2, a3, Args);
  if ( result )
  {
    if ( !check_autotest_var() )
    {
      print2uart((const char **)"first bootm return error, retry use decrypto\n");
      *((_DWORD *)off_2100F8BC + 1) = 0;
      if ( do_decrypto() )
      {
        print2uart((const char **)"retry bootm decrypto return error\n");
        sub_2102CF18(0, 0, 0, 0);
      }
    }
    return bootm_try(a1, a2, a3, Args);
  }
  return result;
```

U-BOOT: DECRYPT KERNEL

```
int do_decrypto()
{
  unsigned __int8 *env_var; // r5
  int v1; // r0
  unsigned __int8 *v2; // r4
  char *v3; // r5
  int v4; // r4
  int allign_size; // r0
  int v6; // r4

  env_var = (unsigned __int8 *)get_env_var("fileaddr");
  v1 = get_env_var("filesize");
  if ( !env_var )
  {
    print2uart((const char **)"do_decrypto addr = NULL\n");
    return 0xFFFFFFFF;
  }
  v2 = (unsigned __int8 *)v1;
  v3 = (char *)atoi(env_var, 0, 0x10u);
  if ( !v2 )
  {
    print2uart((const char **)"do_decrypto size = NULL\n");
    return 0xFFFFFFFF;
  }
  v4 = atoi(v2, 0, 0x10u);
  allign_size = get_allign_size(v4);
  v6 = block_aes_decrypt(v3, v3, allign_size + v4);
  if ( v6 )
    print2uart((const char **)"do_decrypto block_aes_decrypt error\n");
  return v6;
```

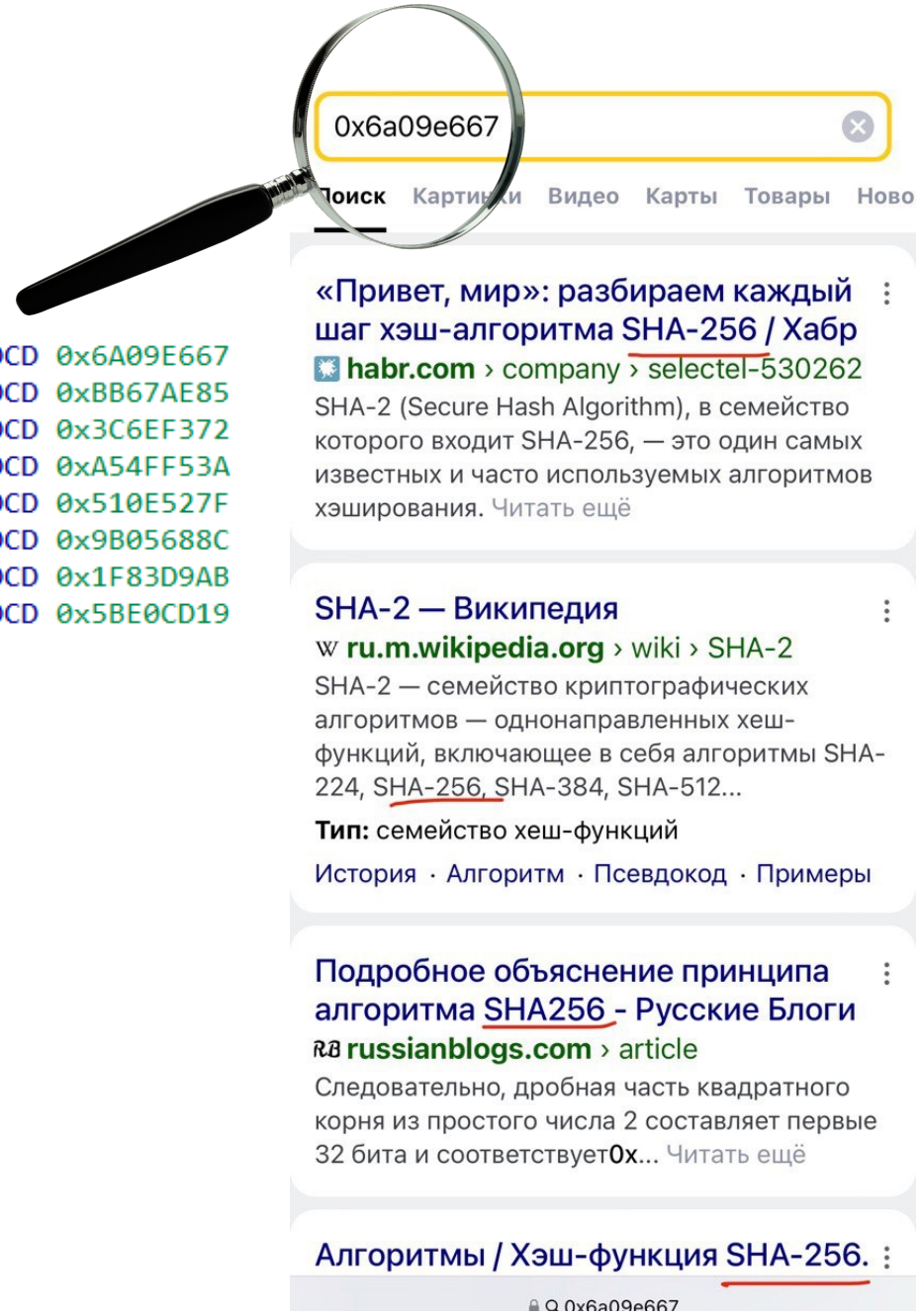
U-BOOT: DECRYPT KERNEL

```
int __fastcall block_aes_decrypt(char *dst_addr, char *src_addr, unsigned int size)
{
  int bl_cnt; // [sp+14h] [bp-24h] BYREF
  char key[32]; // [sp+18h] [bp-20h] BYREF

  memset(key, 0, 0x10);
  bl_cnt = 0;
  if ( firmware_get_key(key) )
    print2uart("firmware_get_key is faild !\n");
  if ( aes_setkey(key, 0x10, 0) )
    print2uart("set key factors error !\n");
  if ( (aes_process(src_addr, size, dst_addr, size, 0x10u, 1, &bl_cnt) & 0x80000000) != 0 )
    print2uart("u-boot encrypt file\n");
  return 0;
```

U-BOOT: FW KEY

```
int __fastcall firmware_get_key(char *key_result)
{
  _BYTE *len_ssc325; // r0
  sha256_calc ctx; // r0
  sha256_calc ctx_; // r0
  int v5; // r2
  char *v6; // r2
  int cnt; // r3
  char last_byte; // t1
  char calced_sha[32]; // [sp+0h] [bp-198h] BYREF
  int cur_sha_inst[11]; // [sp+20h] [bp-178h] BYREF
  char data4sha[256]; // [sp+90h] [bp-108h] BYREF

  memset(calced_sha, 0, sizeof(calced_sha));
  memcpy_0(data4sha, ptr_data4sha, sizeof(data4sha));
  len_ssc325 = strlen("ssc325");
  memcpy_0(data4sha, "ssc325", len_ssc325);
  sha256_starts(cur_sha_inst);
  ctx.sha_cur_val = cur_sha_inst;
  ctx.sha_data = data4sha;
  sha_update(ctx, 0x100u);
  ctx_.sha_cur_val = cur_sha_inst;
  ctx_.sha_data = calced_sha;
  sha_digest(ctx_, v5);
  v6 = cur_sha_inst;
  for ( cnt = 0; cnt != 0x10; ++cnt )
  {
    last_byte = *--v6;
    key_result[cnt] = last_byte ^ calced_sha[cnt];
  }
```

```
dword_21021F20  DCD 0x6A09E667
dword_21021F24  DCD 0xBB67AE85
dword_21021F28  DCD 0x3C6EF372
dword_21021F2C  DCD 0xA54FF53A
dword_21021F30  DCD 0x510E527F
dword_21021F34  DCD 0x9B05688C
dword_21021F38  DCD 0x1F83D9AB
dword_21021F3C  DCD 0x5BE0CD19
```



U-BOOT: FW KEY

```
int __fastcall firmware_get_key(char *key_result)
{
  _BYTE *len_ssc325; // r0
  sha256_calc ctx; // r0
  sha256_calc ctx_; // r0
  int v5; // r2
  char *v6; // r2
  int cnt; // r3
  char last_byte; // t1
  char calced_sha[32]; // [sp+0h] [bp-198h] BYREF
  int cur_sha_inst[11]; // [sp+20h] [bp-178h] BYREF
  char data4sha[256]; // [sp+90h] [bp-108h] BYREF

  memset(calced_sha, 0, sizeof(calced_sha));
  memcpy_0(data4sha, ptr_data4sha, sizeof(data4sha));
  len_ssc325 = strlen("ssc325");
  memcpy_0(data4sha, "ssc325", len_ssc325);
  sha256_starts(cur_sha_inst);
  ctx.sha_cur_val = cur_sha_inst;
  ctx.sha_data = data4sha;
  sha_update(ctx, 0x100u);
  ctx_.sha_cur_val = cur_sha_inst;
  ctx_.sha_data = calced_sha;
  sha_digest(ctx_, v5);
  v6 = cur_sha_inst;
  for ( cnt = 0; cnt != 0x10; ++cnt )
  {
    last_byte = *--v6;
    key_result[cnt] = last_byte ^ calced_sha[cnt];
  }
```

```
#########################################################
data4sha = bytearray(open(os.path.join(KEY_FOLDER,"sha_init_data.bin"), "rb").read())
ciphertext = open(file4decrypt, "rb").read()  # change namr file for decrypt
secure_str = b"ssc325"
src_key_array = bytearray(open(os.path.join(KEY_FOLDER,"aesFactors.bin"), "rb").read())
#########################################################

for idx, val in enumerate(secure_str):
    data4sha[idx] = val

m = hashlib.sha256()
m.update(data4sha)
sha_result = m.digest()
print("sha256:", sha_result)
print("sha256_half1:", sha_result[:0x10])
print("sha256_half2:", sha_result[0x10:0x20][::-1])
fw_key = bytes(a^b for (a, b) in zip(sha_result[:0x10], sha_result[0x10:0x20][::-1]))
print("Fw key:",fw_key)
```

U-BOOT: AES KEY

```
unsigned int __fastcall aes_setkey(char *key, int fw_key_len
{
  char *slot_offset; // r7

  if ( !key )
    return 0xFFFFFFFF;
  if ( (fw_key_len - 1) > 0xF )
    return 0xFFFFFFFF;
  if ( slot > 2 )
    return 0xFFFFFFFF;
  slot_offset = (0x210D1D90 + 0x20 * slot);
  if ( set_slot_val_by_default(slot_offset, 0x20u, slot) )
    return 0xFFFFFFFF;
  memcpy_1(slot_offset, key, fw_key_len);
  return 0;
```

```
g_slot1_setup   DCD 0x37, 0xCC, 0x26C, 0x29E, 0x398, 0x2C, 0x193, 0x195, 0xAD, 0x3B2, 0x66, 0x334, 0x34B, 0x12F, 0x3C4, 0x77, 0x33
                                ; DATA XREF: set_slot_val_by_default:loc_2103A536↑o
                                ; set_slot_val_by_default+5E↑o ...
                DCD 0x271, 0x16, 0x1A6, 0x4D, 0x21, 0x3A6, 0x20A, 0x92, 0x185, 0x251, 0x276, 0x3A7, 0x3D7, 0x1E3, 4; 0x11
g_slot0_setup   DCD 0x375, 0x282, 0xD9, 0x269, 0x381, 0x156, 0x17A, 0x2EC, 0x160, 0xB5, 0x75, 0x257, 0x3A9, 0x39B, 0x7A, 0x3A7, 0x
                                ; DATA XREF: set_slot_val_by_default:loc_2103A52E↑o
                                ; set_slot_val_by_default+56↑o ...
                DCD 0x172, 0x30B, 0x305, 0x232, 0x27B, 0xD3, 0x14C, 0x1C2, 0x28A, 0x151, 0x122, 0x15C, 0x15B, 0x284, 0x352; 0x11
g_slot2_setup   DCD 0x33F, 0x360, 0x163, 0xE0, 0x2AA, 0x219, 3, 0x33D, 0x286, 0x17C, 0x35, 0x7D, 0x15, 0x172, 0x23F, 0xB7, 0x14, 0
                                ; DATA XREF: set_slot_val_by_default+4A↑o
```

```
int __fastcall set_slot_val_by_default(char *slot_offset, unsigned int size, int slot)
{
  int v6; // r8
  int *struct_for_slot; // r8
  int cnt; // r4
  int i; // r3
  int j; // r4
  char IV[16]; // [sp+Ch] [bp-10h] BYREF
  char src_ptr[32]; // [sp+1Ch] [bp+0h] BYREF
  char Key[32]; // [sp+3Ch] [bp+20h] BYREF
  int exp_key[61]; // [sp+5Ch] [bp+40h] BYREF

  memset(src_ptr, 0, sizeof(src_ptr));
  if ( !slot_offset || size <= 0x1F )
    return 0xFFFFFFFF;
  memset(Key, 0, sizeof(Key));
  memset(exp_key, 0, 0x26);
  if ( slot )
  {
    if ( slot != 1 )
    {
      if ( slot == 2 )
      {
        struct_for_slot = g_slot2_setup;
        goto LABEL_10;
      }
      return 0xFFFFFFFF;
    }
    struct_for_slot = g_slot1_setup;
  }
  else
  {
    struct_for_slot = g_slot0_setup;
  }
```

U-BOOT: AES KEY

```
do
{
  src_ptr[cnt] = InterArrayy[cnt] + 1 + ((struct_for_slot[cnt] + 0x258) >> 1);
  ++cnt;
}
while ( cnt != 0x20 );
sub_2103A484((unsigned int)exp_key, (int *)Key);
v6 = 0;
memset_0(Key, 0, sizeof(Key));
memset_0(exp_key, 0, 0x26);
for ( i = 0; i != 0x10; ++i )
  Key[i] = i + 0x20;                            // make KEY for decrypt KEY_AES_FW
for ( j = 0; j != 0x10; ++j )
  IV[j] = 0x65;                                 // make IV
do
{
  Key[j] = j + 0x13;                            // make KEY for decrypt KEY_AES_FW
  ++j;
}
while ( j != 0x20 );
if ( (expand_key((unsigned __int8 *)Key, 0x100, exp_key) & 0x80000000) == 0 )// expand aes key
{
  encrypt_aes(exp_key, (int)src_ptr, 0x20, IV, slot_offset);// decrypt KEY_AES_FW
  memset_0(exp_key, 0, sizeof(exp_key));
  memset_0(Key, 0, sizeof(Key));
  memset_0(IV, 0, sizeof(IV));
}
```

```
DCB 0xC4, 0x58, 0xDB, 0x30, 0x48, 0xF4, 0x78, 0x19, 0x9E, 5, 0x74, 0xF3, 0xB7, 0x24, 0x45, 0x75, 0xB6, 3, 0x93, 0x18;
                        ; DATA XREF: set_slot_val_by_default:loc_2103A53C↑o
                        ; set_slot_val_by_default+68↑o ...
DCB 0x34, 0xA3, 0x42, 0x4F, 0x3C, 0x6C, 0xE0, 0x14, 0xF, 0x79, 0x26, 0x42; 0x14
```

U-BOOT: AES KEY

```python
def make_key_slot(src_key_array, src_slot_array):
    print("Start generate key slot data:-------")
    key = bytearray(0x20)
    for i in range(0x10):
        key[i] = (i+0x20)
    for i in range(0x10, 0x20):
        key[i] = (i+0x13)
    print("key:", key)
    iv = 0x10*b"\x65"
    print("iv:", iv)
    data = bytearray(0x20)
    for i in range(0x20):
        tmp1 = struct.unpack("<B", src_key_array[i:i+1])[0]
        tmp2 = ((struct.unpack("<L", src_slot_array[i*4:i*4+4])[0] + 0x258) >> 1)
        data[i] = (tmp1 + 1 + tmp2) & 0xFF
    print("data:", data)
    ciphertext = aes_encrypt(data, key, iv)
    print("ciphertext:", ciphertext)
    print("End generate key slot data:-------")
    return ciphertext
```

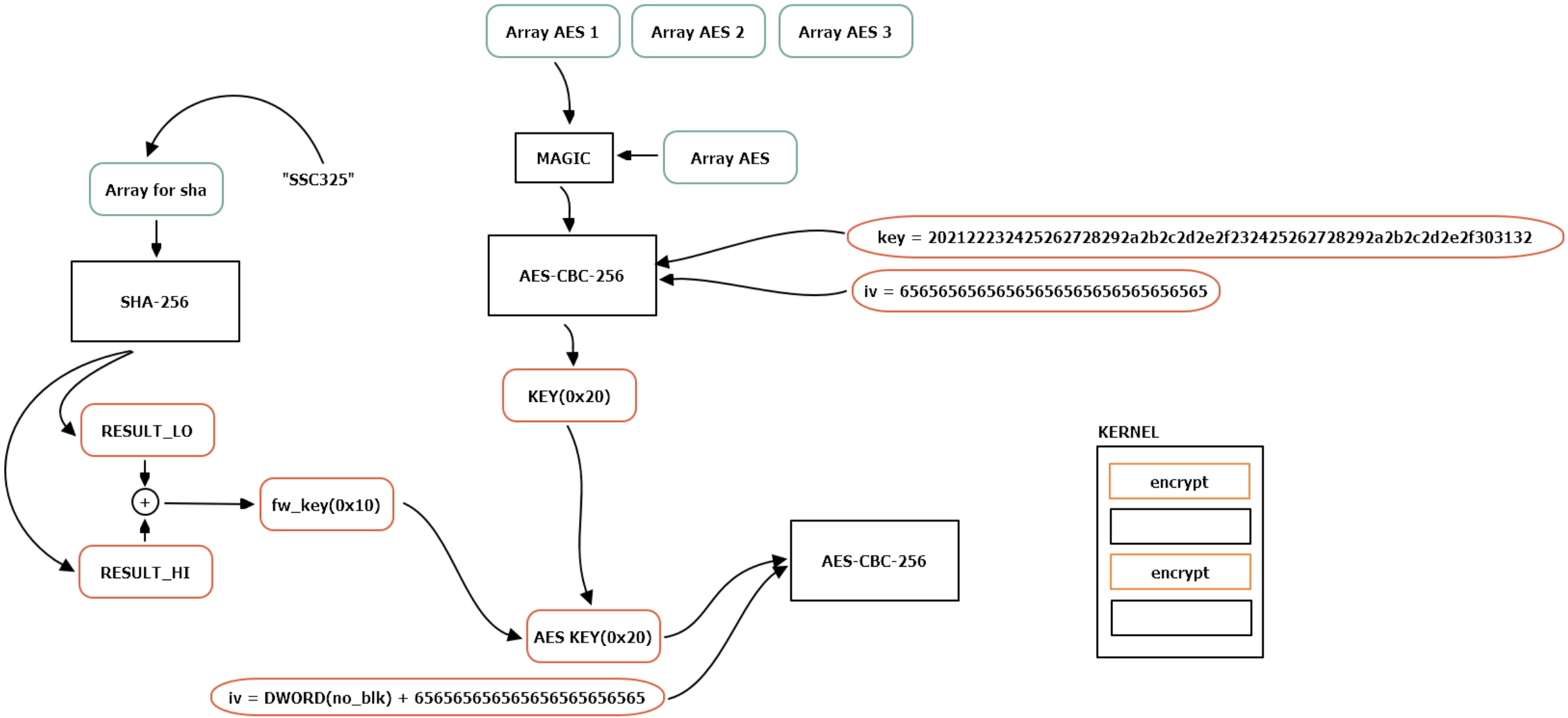
```python
key = bytearray(make_key_slot(src_key_array, src_slot_array))
for i, v in enumerate(fw_key):
    key[i] = v
print("AES key:",key)
iv = get_iv(0)
```

U-BOOT: AES PROCESS

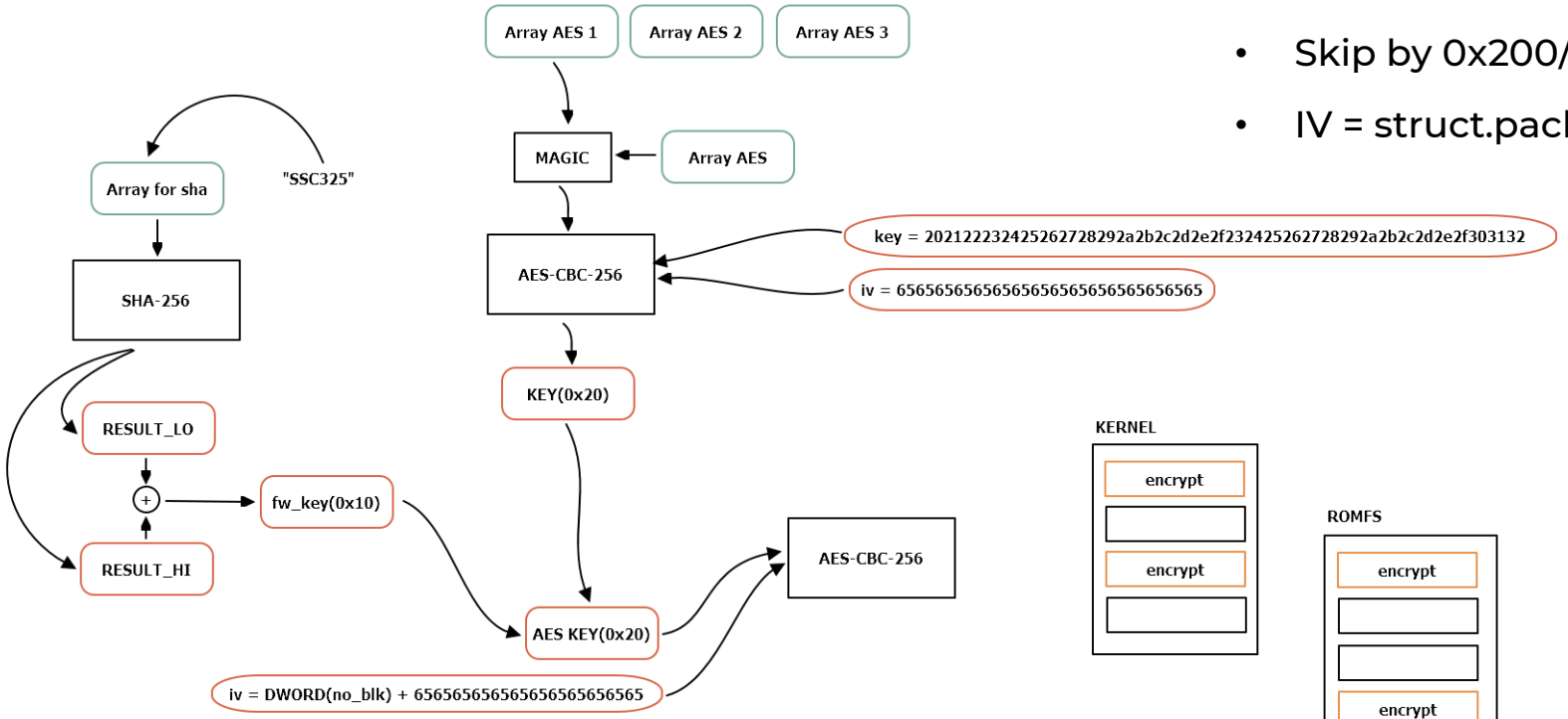
```
tmp_rc_addr = src_addr;
decrypted_size = 0;
step_bytes = CntBlock << 9;
while ( decrypted_size < src_size )
{
  if ( src_size - decrypted_size <= 0x1FF )
  {
    if ( one == 1 )
      return 0xFFFFFFFF;
    memcpy_1(v25, &src_addr[decrypted_size], src_size - decrypted_size);
    tmp_rc_addr = v25;
  }
  src_ptr = (unsigned int)tmp_rc_addr;
  memset_0(IV, 0, sizeof(IV));
  memcpy_1(IV, (char *)&aes_bl_round, 4);

  dst_ptr = (unsigned __int8 *)&dst_addr[decrypted_size];

  if ( one )
    decrypt_aes(expand_key, src_ptr, 0x200, IV, dst_ptr);
  else
    encrypt_aes(expand_key, src_ptr, 0x200, IV, (char *)dst_ptr);

  decrypted_size += 0x200;
  if ( CntBlock )
  {
    if ( src_size <= decrypted_size )
      break;
    v16 = &dst_addr[decrypted_size];
    v17 = &src_addr[decrypted_size];
    if ( step_bytes >= src_size - decrypted_size )
      step_bytes = src_size - decrypted_size;
    decrypted_size += step_bytes;
    memcpy_1(v16, v17, step_bytes);          // расшифровка через блок
  }
  tmp_rc_addr = &src_addr[decrypted_size];
  ++aes_bl_round;
```

```
aes_bl_round = 0
plaintext = bytearray()
print("Decrypting.....")
AES_BLOCK_SIZE = 0x200
for i in range(0, len(ciphertext), AES_BLOCK_SIZE + skip_decrypt_bytes):
    aes_bl_end = i+AES_BLOCK_SIZE
    data1 = ciphertext[i:aes_bl_end]
    data2 = ciphertext[aes_bl_end:aes_bl_end+skip_decrypt_bytes]
    plaintext.extend(aes_decrypt(data1, key, get_iv(aes_bl_round)))
    plaintext.extend(data2)
    aes_bl_round += 1
print("Decrypting DONE")
open("result_aes", "wb").write(plaintext)
```

U-BOOT: AES KEY



KERNEL: DECRYPT ROMFS



- AES-CBC 256 bits
- Skip by 0x200/0x400 bytes
- IV = struct.pack("<L", no_blk) + 12 * b'\x00'

- aesFactors.bin
- keyEnvHand.bin
- keyKernelHand.bin
- keyRootfsHand.bin
- sha_init_data.bin

THEIR MANNERS

SSH password: 7ujMko0 + <password_cam>

```
v34 = strncmp("7ujMko0", (const char *)&Password, 7u);
if ( !v34 )
{
  v35 = s1;
  _snprintf_chk(JustPass, 0x80, 1);
  BYTE2(v74) = 0;
  LOWORD(v74) = 0;
```

Encryption key for config file

```
if ( sub_6B538(a4, "VTHRemoteIPCInfo") )
  v16 = "DHRDENFR";
else
  v16 = "MWPZWJGS";
v17 = *v16;
```

https://github.com/mcw0/Tools/blob/master/DahuaConfigBackupDecEnc.py

QR CODE

SSH connect:

```
root@        ~# ssh admin@192.168.1.108
admin@192.168.1.108's password:
JSON Parse Failed, Try again
JSON Parse Failed, Try again
Date&Time: Jul 17 2020 10:16:25
Revision: 80062
Enter 'help' for a list of commands (dsh)

#help

Support Commands:

shell                          help                          getDateInfo
diagnose
Please set UTF-8 character encoding format in terminal for displaying Qrcode
#
```

help &
getDateInfo cmd:

```
#help

Support Commands:

shell                          help                          getDateInfo
diagnose
Please set UTF-8 character encoding format in terminal for displaying Qrcode
#getDateInfo

Sat Feb 1  0:19:50 UTC 2000
```

QR CODE

CMD: shell

https://svsh.dah6.com/svsh.html?

v=2&

u=1&

t=FZdfNVEoz0i5yZjTC1aFGHn8H3Y%2BEQEFB%2Fuuv4G%2F60sgPrliQfVGQzDjTFQWNKrQ

aSSr6PDFvqHm74RCXD%2BqaOegivbqKKAgI3PsXObXUcuBAnpctF%2F8FO1QFjf8jz3%2F

nF5nEL3AgXvocxtSHzBVCaCaF3T35mcPKic358iagHllc%2FLYtUBkROmnKafRjT5K

UJJNcC5txHlUJS0tLtr%2BuQHeKerpTml5nUSAgLFOat8yr7C5%2B3N%2FE1ODYv%2B%2Fj41D

duP1qmJzN0mPBs%2FvyOuJL%2F7zJYRs4J6PC6lGo5I%2FezGyT6X6TnYVKC84rR9kJ0Zz

CAmtT4B9AHIpCAeLFiCJxg%3D%3D

V – always 2

U - Domain Accounts

T – RSA_encrypt(QR_struct)

QR CODE

QR code structure generation:

```
outBuf->DomainAccNo = Domain_Accounts;
outBuf->two = 2;
outBuf->MAC_H = *(_DWORD *)MAC;
outBuf->MAC_L = *(_WORD *)(MAC + 4);
outBuf->gCurTime = gCurTime;
result = rand32();
outBuf->selfPtr = (int)outBuf;
serial = (int *__shifted(QR_GENCODE,0x12))outBuf->serial;
v9 = SerialID;
ADJ(serial)->random = result;
do
{
  tmp = *v9++;
  *serial++ = tmp;
}
while ( v9 != SerialID + 3 );
*(_WORD *)serial = *(_WORD *)v9;
ADJ(serial)->serial[2] = *((_BYTE *)v9 + 2);
```

QR structure:

- Two
- DomainAccNo
- MAC_H
- MAC_L
- gCurTime
- Serial[15]
- Rand32
- selfPtr
- Empty_field_29

QR CODE

URL generation:

```
rsa_encode_pub(qr_struct, (int)enc_buff, encode_len),
encode_len_1 = sub_14414((int)enc_buff, encode_len, enc_buff_2, 2 * encode_len),
len = snprintf(
        (char *)url_s,
        v10,
        "https://svsh.dah6.com/svsh.html?v=%d&u=%d&t=",
        qr_struct->two,
        qr_struct->DomainAccNo),
!url_encode((const char *)enc_buff_2, encode_len_1, (int)url_s + len, v10 - len)) )
```

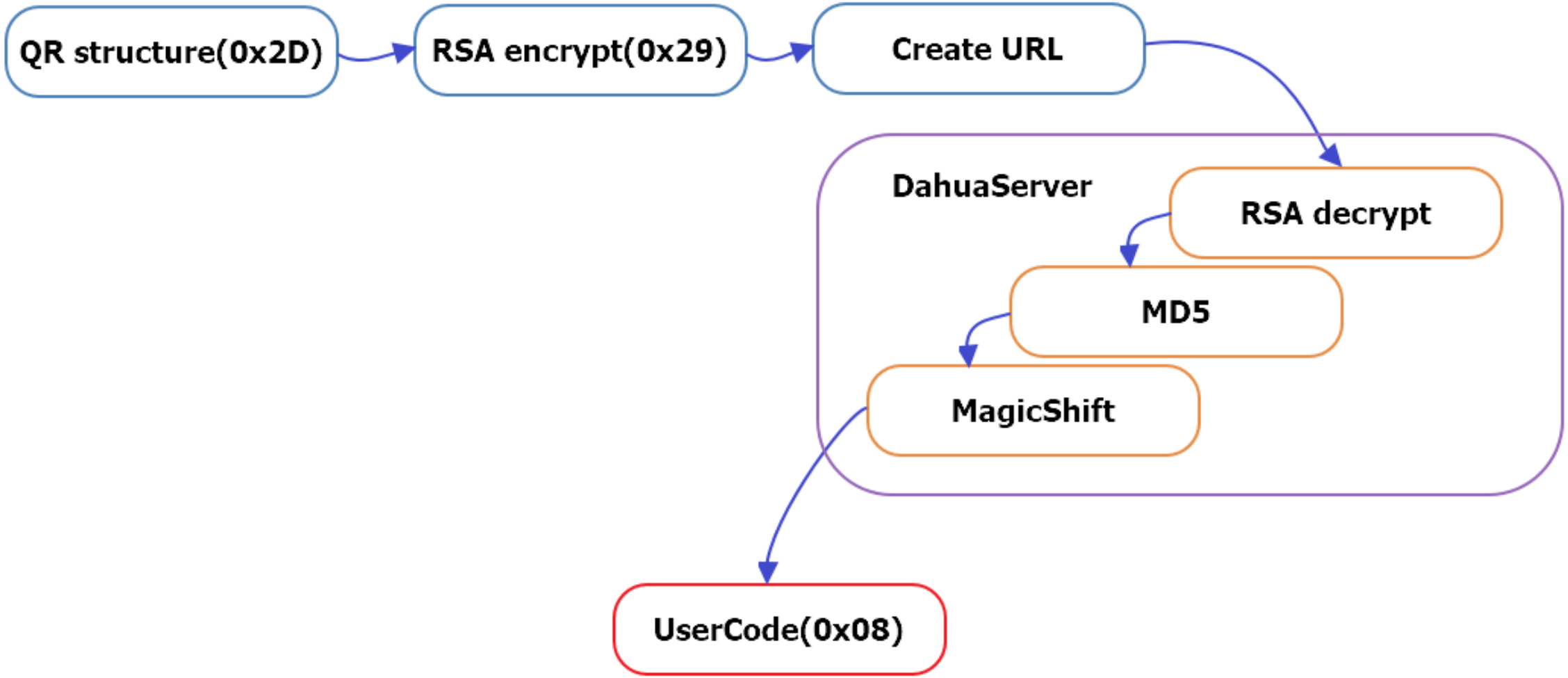
- RSA encrypt just 0x29 bytes
- RSA-2048
- Random padding
- Create URL
- Create QR code

Verify user code:

```
bool __fastcall VerifyCode(QR_GENCODE *gQrGen, const void *pass)
{
  int DomainAccNo; // r5
  unsigned int shift; // r1
  char ascii_md5_hash[33]; // [sp+4h] [bp-34h] BYREF

  memset(ascii_md5_hash, 0, sizeof(ascii_md5_hash));
  if ( !gQrGen || !pass || gQrGen != (QR_GENCODE *)gQrGen->selfPtr )
    return 0;
  DomainAccNo = gQrGen->DomainAccNo;
  MD5((char *)gQrGen, ascii_md5_hash);
  shift = reverse_bits(DomainAccNo, 0x18u);
  return memcmp(&ascii_md5_hash[shift], pass, 8u) == 0;
```

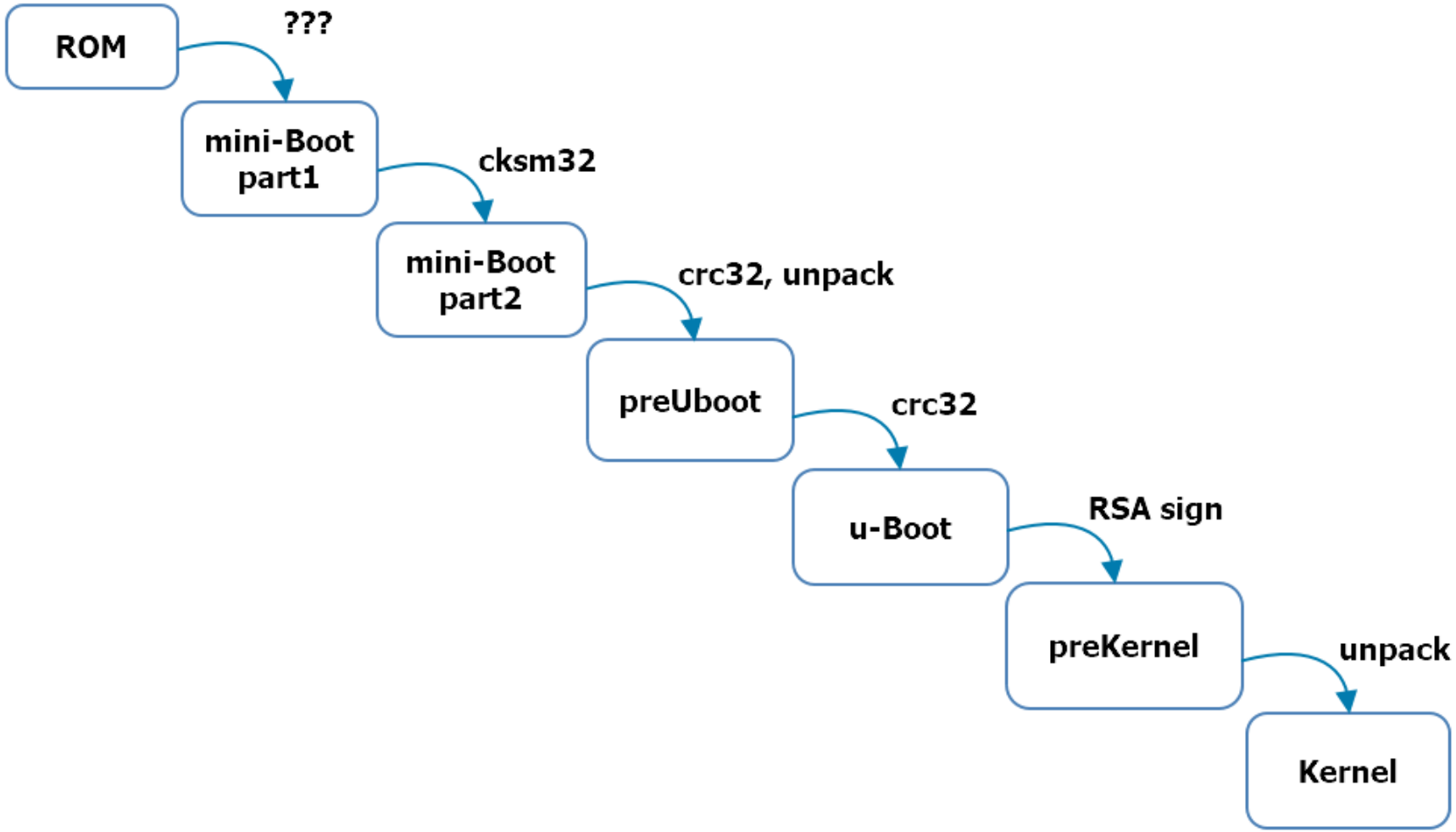
- MD5(QR)
- Shift = Magic(DomainAcc)
- md5_result[shift] == user_code

QR CODE

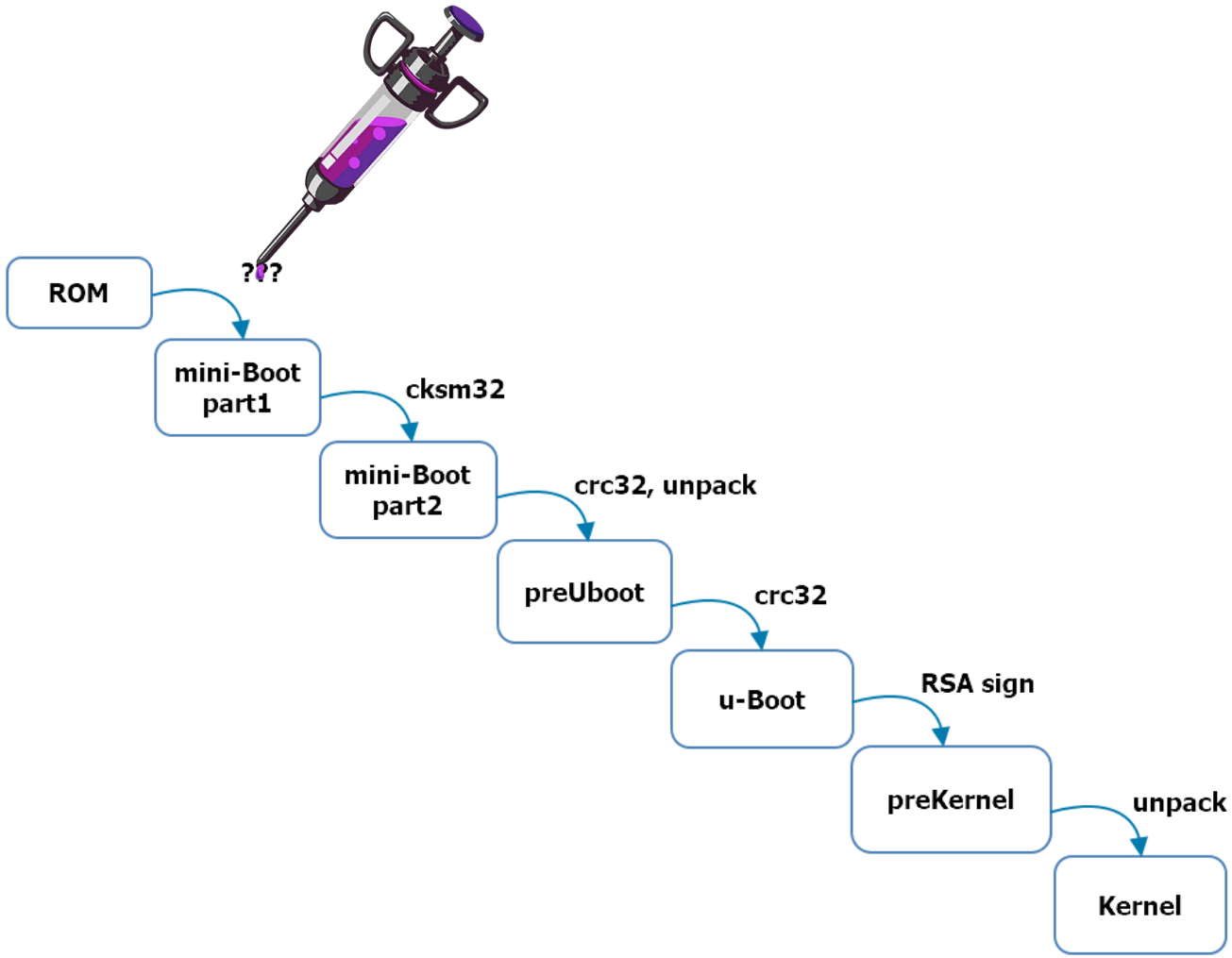

QR structure(0x2D)
RSA encrypt(0x29)
Create URL
DahuaServer
RSA decrypt
MD5
MagicShift
UserCode(0x08)

ROOT OF TRUST?

ROOT OF TRUST



ROOT OF TRUST



UART log:

```
IPL fd8e347
D-0a
517MB
BIST0_0001-OK
Load IPL_CUST from SPINAND
CIS in Block00
Match CIS ID( c8 01 )

 BlSize 00004780
Checksum OK

IPL_CUST fd8e347
runEBOOT()
CIS in Block00
Match CIS ID( c8 01 )
ChkHealthy: 0003: 0007: 0000
ChkHealthy: 0004: 0007: 0000
ChkHealthy: 0005: 0007: 0000

Load BL from SPINAND
```

patch ver 5

UART debug:

```
Load BL from SPINAND
 -Verify CRC32 failed!
calc_crc32=0xe2e0bc21
 header dcrc=0xe2e0bc21
```

The End